



NAME: VOMMIDAPU CHINNI

BATCH NO: 120

MAIL.ID: vommidapuchinni@gmail.com

DATE: 19/03/24

TRAINER: Mr. Madhukar

MODULE: Terraform

ASSIGNMENT – 3

1. By using terraform create an s3 bucket and upload the file to the s3 bucket?

To create an S3 bucket and upload files to it using Terraform, you can follow these steps:

1. Launch an EC2 instance.
2. Create a user in IAM.
3. Connect to the server, Install awscli in server.
4. Install terraform in server.
5. Create terraform directory, Create terraformblock.tf, Create provider.tf, Create resource.tf.

Step1: Launch an EC2 instance

- a. Click on launch instance.
- b. Give name/tag.
- c. Select AMI as ubuntu.
- d. Instance type t2.micro.
- e. Create key pair as terraform-1.

[Launch an instance | EC2 | us-east-1](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances)

PRED National Portal of... DCC Login Nadu Nedu APREIS - 2020 Home APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stu... Home Page/Press In... Home | Endowment... N. Virginia Vommidapu Chinni

EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Recents Quick Start

Summary

Number of instances Info 1

Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-080e1f13689e07408

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which you launch)

Cancel Launch instance Review commands

CloudShell Feedback

Tomorrow's low Near record Search ENG IN 08:46 19-03-2024

[Launch an instance | EC2 | us-east-1](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances)

PRED National Portal of... DCC Login Nadu Nedu APREIS - 2020 Home APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stu... Home Page/Press In... Home | Endowment... N. Virginia Vommidapu Chinni

EC2 > Instances > Launch an instance

Launch an instance Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Recents Quick Start

Application and OS Images (Amazon Machine Image) Info

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Li Browse more AMIs Including AMIs from AWS Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type Free tier eligible
ami-080e1f13689e07408 (64-bit (x86)) / ami-0a55ba1c20b74fc30 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2024-03-01

Architecture AMI ID

Summary

Number of instances Info 1

Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-080e1f13689e07408

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

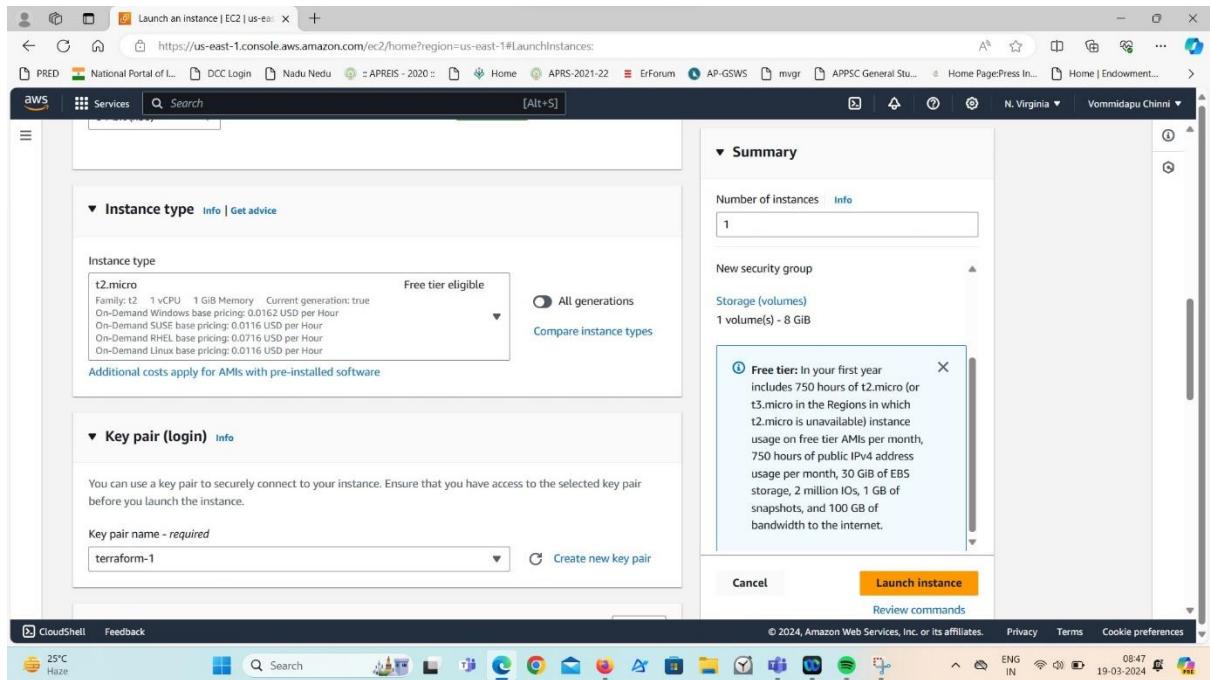
Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which you launch)

Cancel Launch instance Review commands

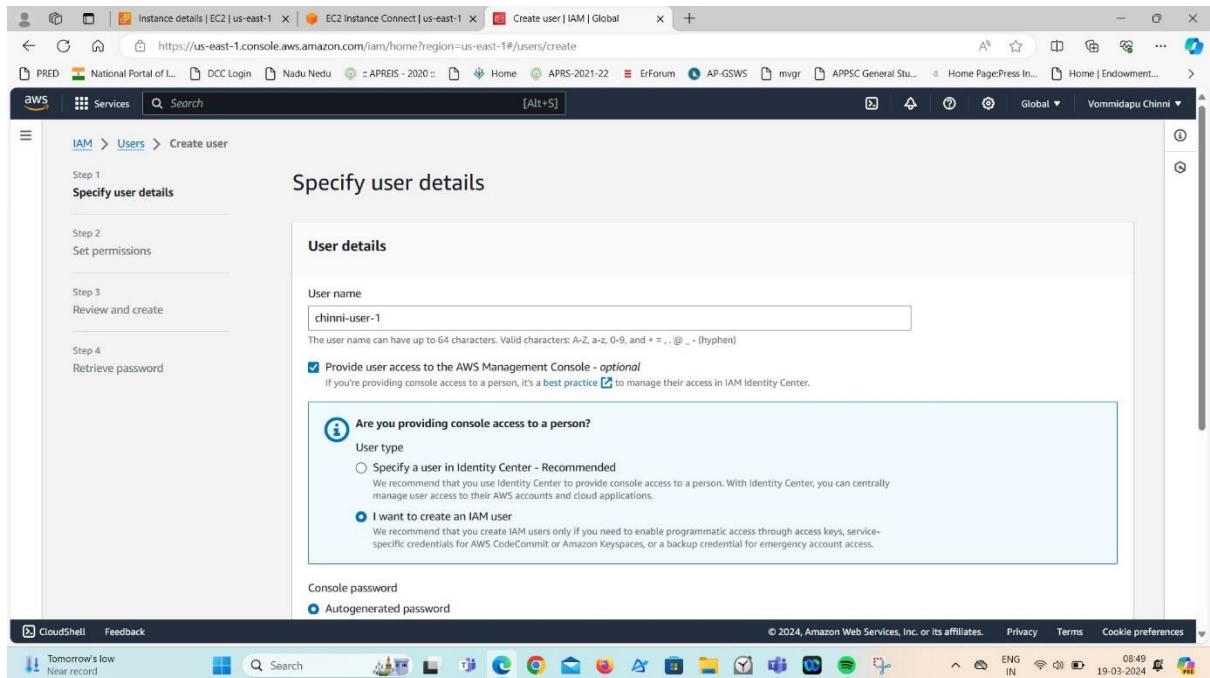
CloudShell Feedback

Tomorrow's low Near record Search ENG IN 08:46 19-03-2024



Step2: Create user IAM.

- Search IAM, open it.
- In dashboard click on user.
- Click on create user.
- Give user name.
- Click on Attach policies directly, give s3 full access.
- Create user.



Instance details | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | Create user | IAM | Global

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create

PRED National Portal of... DCC Login Nadu Nedu APREIS - 2020 APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stu... Home Page/Press In... Home | Endowment... Global Vommidapu Chinni

aws Services Search [Alt+S]

Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keypairs, or a backup credential for emergency account access.

Console password
 Autogenerated password
You can view the password after you create the user.
 Custom password
Enter a custom password for the user.

Must be at least 8 characters long.
Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # \$ % ^ & * () _ + - (hyphen) = [{ }]

Show password

Users must create a new password at next sign-in - Recommended
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user. [Learn more](#)

Cancel Next

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Haze Search

ENG IN 08:49 19-03-2024

Instance details | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | Create user | IAM | Global

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create

PRED National Portal of... DCC Login Nadu Nedu APREIS - 2020 APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stu... Home Page/Press In... Home | Endowment... Global Vommidapu Chinni

aws Services Search [Alt+S]

IAM > Users > Create user

Step 1 Specify user details

Step 2 Set permissions

Step 3 Review and create

Step 4 Retrieve password

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (2/1199)

Choose one or more policies to attach to your new user.

Filter by Type: s

Policy name	Type	Attached entities
AmazonDMSRedshiftS3Role	AWS managed	0
AmazonS3FullAccess	AWS managed	0

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Haze Search

ENG IN 08:51 19-03-2024

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name chinni-user-1	Console password type Autogenerated	Require password reset No
----------------------------	--	------------------------------

Permissions summary

Name	Type	Used as
AmazonS3FullAccess	AWS managed	Permissions policy

Tags - optional
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Cancel Previous Create user

- Create an access key:
- Click on user.
 - On right side we can see create access key, click on it.
 - Select cli, give a tag and create it.

Instance details | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | Create access key | IAM | Global

Step 1
Access key best practices & alternatives

Step 2 - optional
Set description tag

Step 3
Retrieve access keys

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

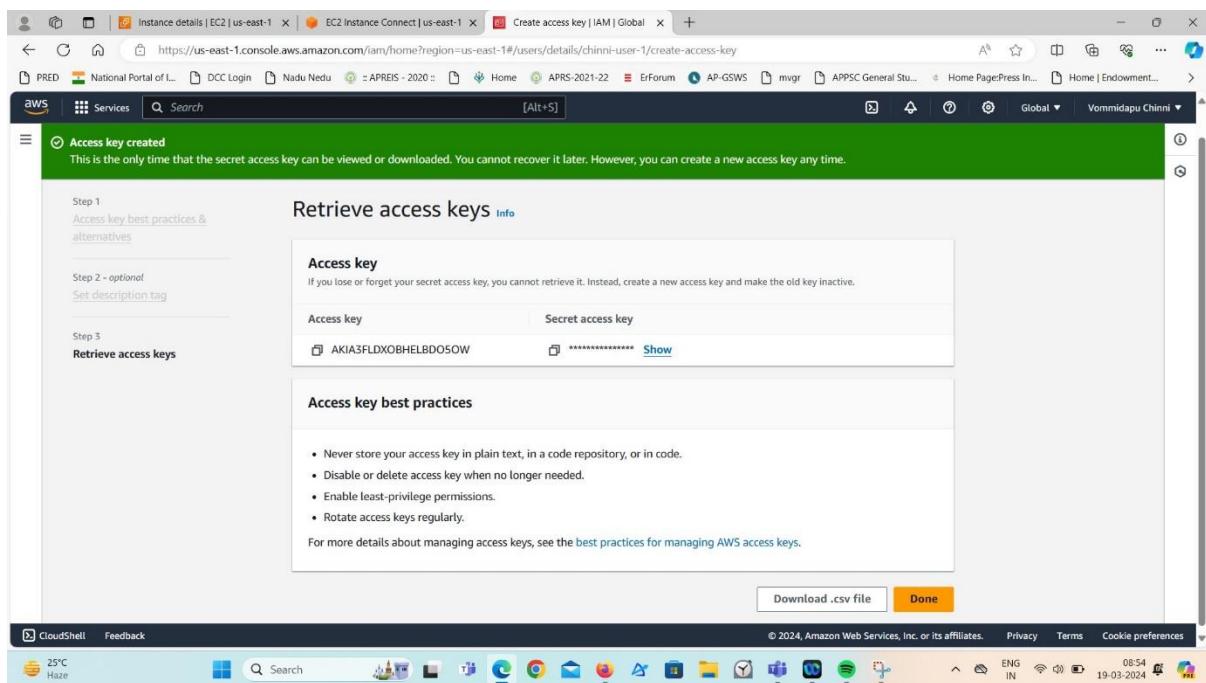
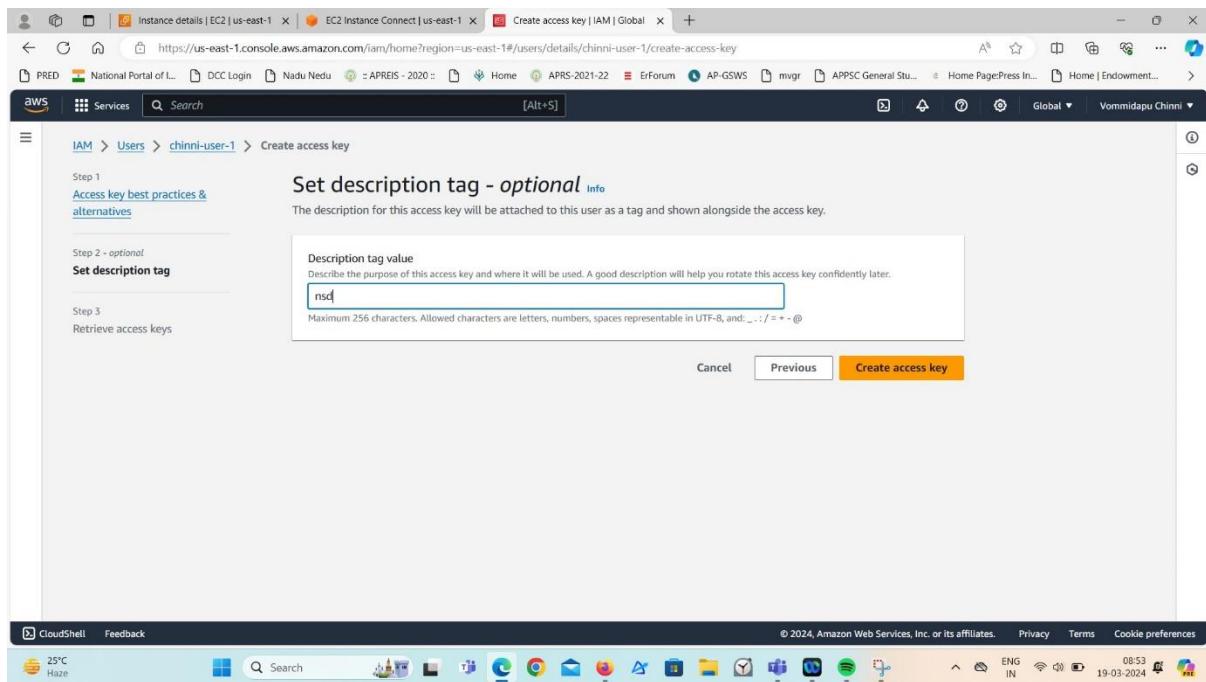
Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Step3: Connect to the server, Install awscli in server

- Click on instance id, open it.
- Click on connect, again click on connect.
- Give commands like:
- Sudo -i (convert normal user to root user).
- apt update -y (to update).
- apt install awscli(install awscli).

- g. aws configure
- h. give access key and secret access key.
- i. Default region us-east-1
- j. Format as table.
- k. ls -a
- l. cd .aws
- m. ls

```
ubuntu@ip-172-31-91-105:~$ sudo -i
root@ip-172-31-91-105:~# apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1490 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [288 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1605 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [268 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1058 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [239 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [22.1 kB]
Get:18 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1246 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [42.1 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [10.1 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [472 B]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [67.1 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.0 kB]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [398 B]

i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Haze 25°C Haze 08:55 19-03-2024 ENG IN

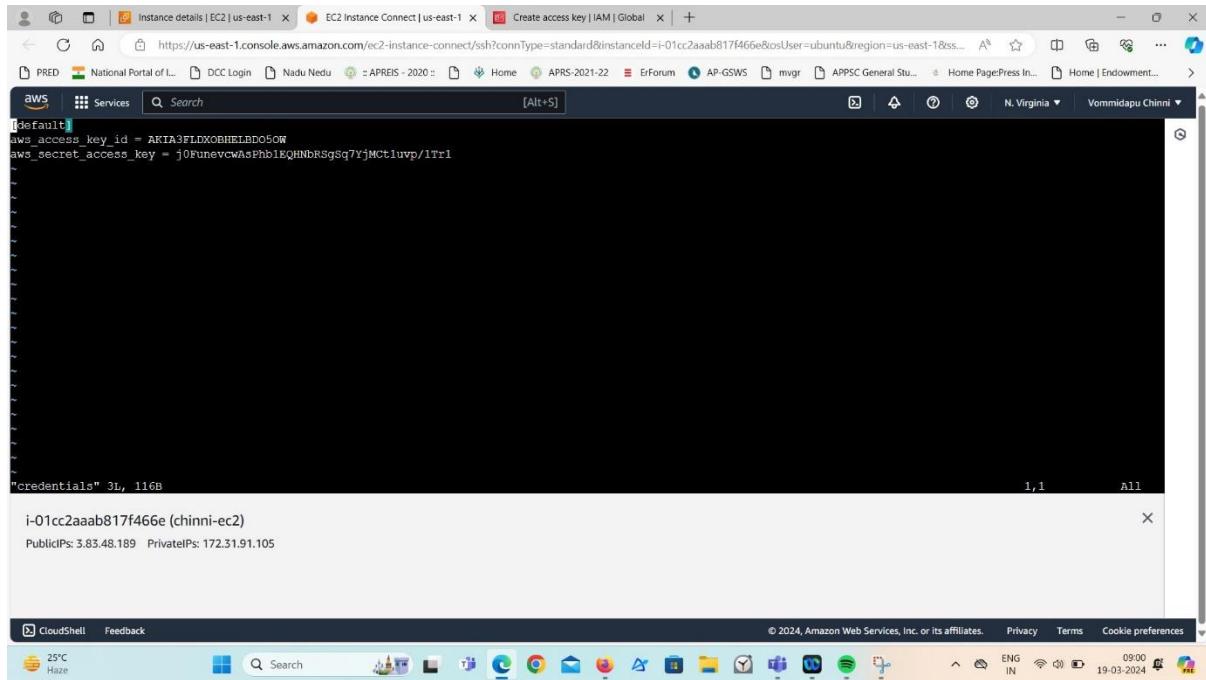
```
ubuntu@ip-172-31-91-105:~$ sudo -i
root@ip-172-31-91-105:~# apt install awscli -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
19 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-91-105:~# apt install awscli -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bzip2 docutils-common fontconfig fontconfig-config fonts-droid-fallback fonts-noto-mono fonts-urw-base35 ghostscript groff gsfonts hicolor-icon-theme imagemagick
  imagemagick-6-common imagemagick-6.q16 libavahi-client3 libavahi-common-data libavahi-common3 libcairo2 libcups2 libdatriel libdavids libde265-0
  libdeflate0 libdjvuibre-text libdjvuibre21 libfftw3-double3 libfontconfig1 libgomp1 libgraphite2-3 libgs9 libgs9-common libharfbuzz0 libheif1 libice6 libidm12
  libjbig-0.35 libimbase25 libimagingquant0 libjbig0 libjbig2dec0 libjpegturbo8 libjpeg9 libjxr-tools libjxr0 libcm2c2 liblqr-1-0 libltdl7 libmagickcore-6.q16-6
  libmagickcore-6.q16-6-extra libmagickwand-6.q16-6 libnetpxm10 libopenexr25 libopenp2-7 libpango-1.0-0 libpangocairo-1.0-0 libpangoft2-1.0-0 libpaper-utils
  libpapercr lippixman-1-0 libraqm0 libsm6 libthai-data libthai10 libtiff7 libwebpdemux2 libwebpmy3 libwmflite-0.2-7 libx265-199 libxaw7 libxcb-render0
  libxcb-shm0 libxmu6 libxpm4 libxrender1 libxt6 maiacmp mime-support netpbm poppler-data psutils python3-botocore python3-dateutil python3-docutils
  python3-jmespath python3-olefile python3-pil python3-pymt python3-pymt python3-roman python3-rsa python3-s3transfer sgml-base x11-common xml-core
Suggested packages:
  bzip2-doc fonts-noto-fonts-freefont-otf | fonts-freefont-ttf fonts-texgyre ghostscript-x imagemagick-doc autotrace cups-bod | lpr | lprng enscript ffmpeg gimp
  gnuplot grada graphviz hp2xx htm2ps libwmf-bin mplayer povray radianc sane-utils texlive-base-bin transfig uraw-batch xdj-utils cups-common libfftw3-bin
  libfftw3-dev libicms2-utils inkscape poppler-utils fonts-japanese-mincho | fonts-ipafont-mincho fonts-japanese-gothic | fonts-ipafont-gothic fonts-aphic-ukai
i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

25°C Haze 25°C Haze 08:57 19-03-2024 ENG IN

vi config:

vi credentials:



```
[Default]
aws_access_key_id = AKIA3FLDXOBHELB050W
aws_secret_access_key = j0FunevcvwAsPh01EQHNbRSgSg7YjMct1uvp/lTrl
~

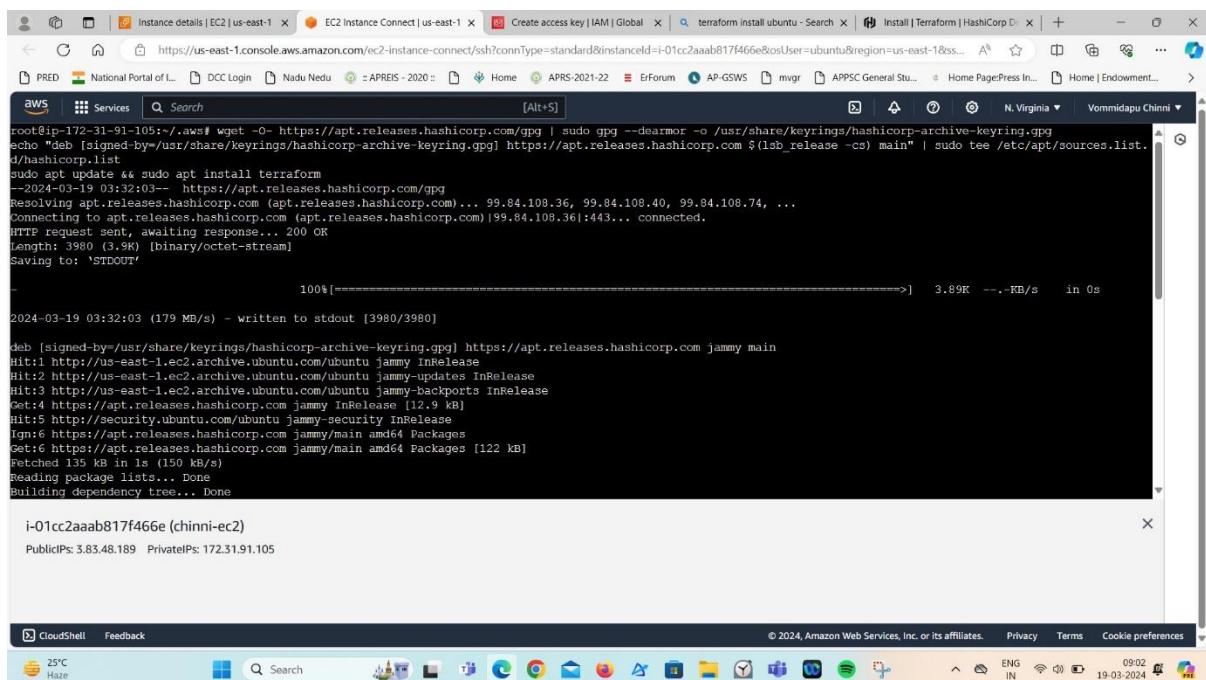


"credentials" 3L, 116B
```

i-01cc2aab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

Step4: Install terraform in server

- By using : wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
- echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com \$(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
- sudo apt update && sudo apt install terraform



```
root@ip-172-31-91-105: ~/.aws# wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform
--2024-03-19 03:32:03--  https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 99.84.108.36, 99.84.108.40, 99.84.108.74, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|99.84.108.36|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

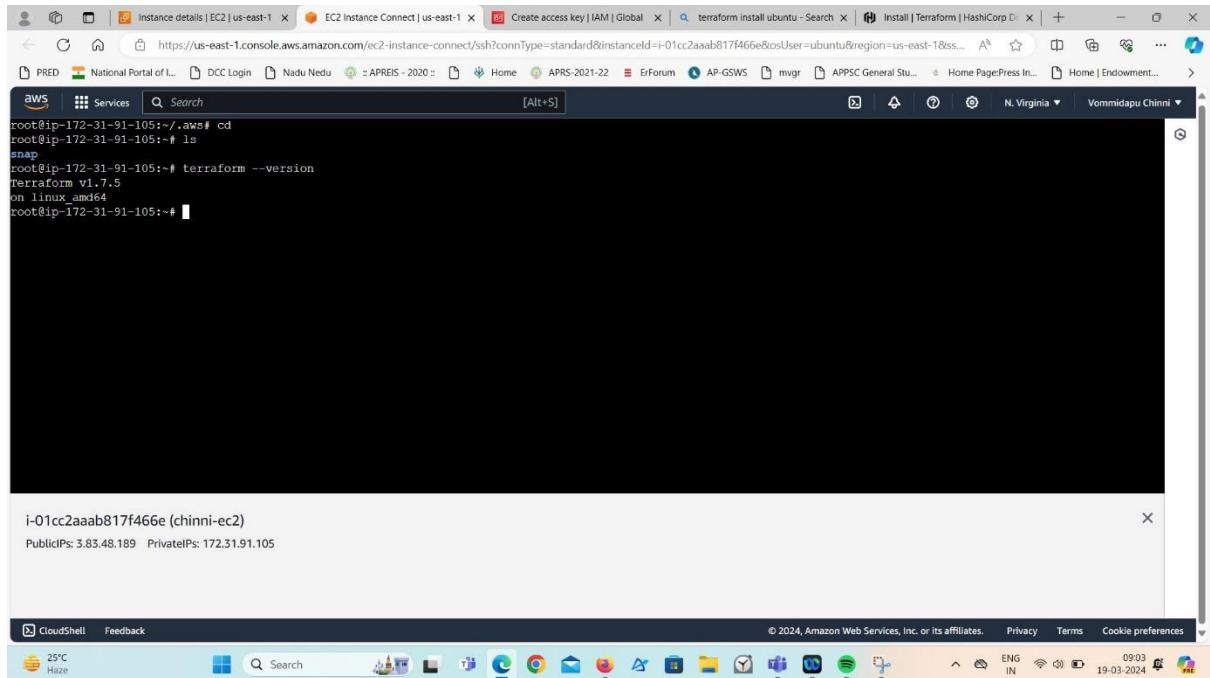
100%[=====] 3.89K --.-KB/s   in 0s

2024-03-19 03:32:03 (179 MB/s) - written to stdout [3980/3980]

deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com jammy main
Hit:1 http://us-east-1.ec2.archive/ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive/ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive/ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://apt.releases.hashicorp.com jammy InRelease [12.9 kB]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Ign: https://apt.releases.hashicorp.com jammy/main amd64 Packages
Get:6 https://apt.releases.hashicorp.com jammy/main amd64 Packages [122 kB]
Fetched 135 kB in 1s (150 kB/s)
Reading package lists... Done
Building dependency tree... Done
Building dependency tree... Done

i-01cc2aab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

- cd (use to change directory)
- terraform –version (to know terraform version).



```
aws Services Search [Alt+S]
root@ip-172-31-91-105:~/.aws# cd
root@ip-172-31-91-105:~# ls
snap
root@ip-172-31-91-105:~# terraform --version
Terraform v1.7.5
on linux_amd64
root@ip-172-31-91-105:~#
```

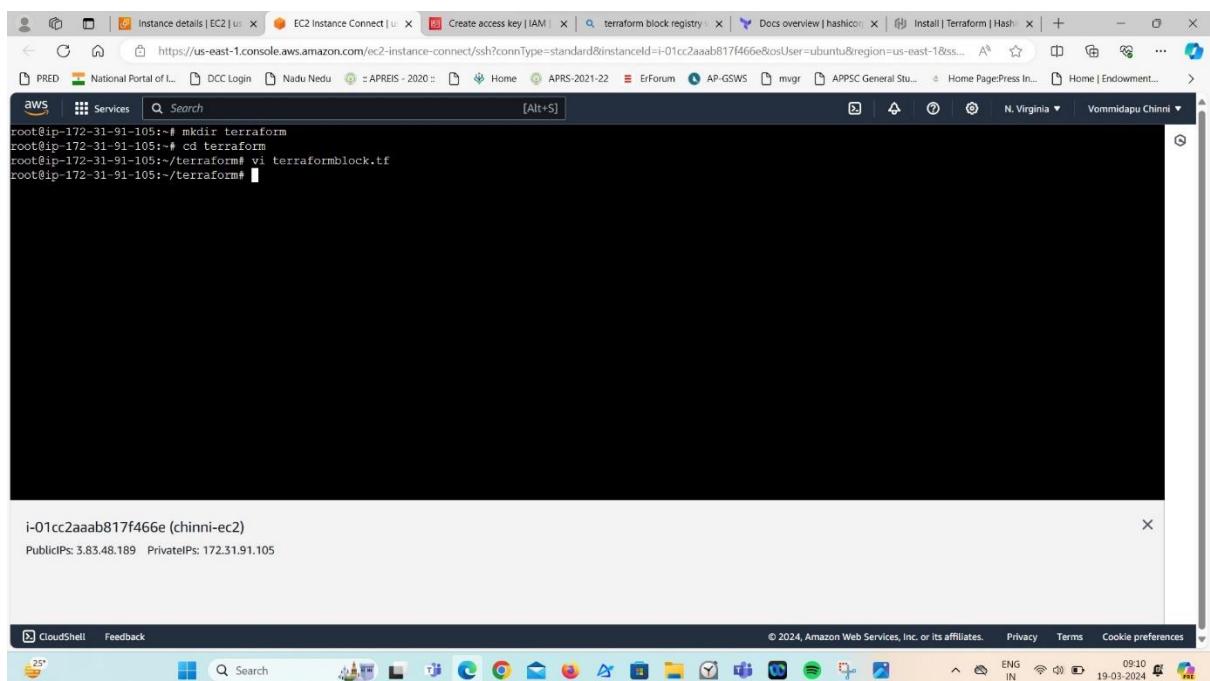
i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
25°C Haze 09:03 ENG IN 19-03-2024

Step5: Create terraform directory:

- mkdir terraform
- cd terraform

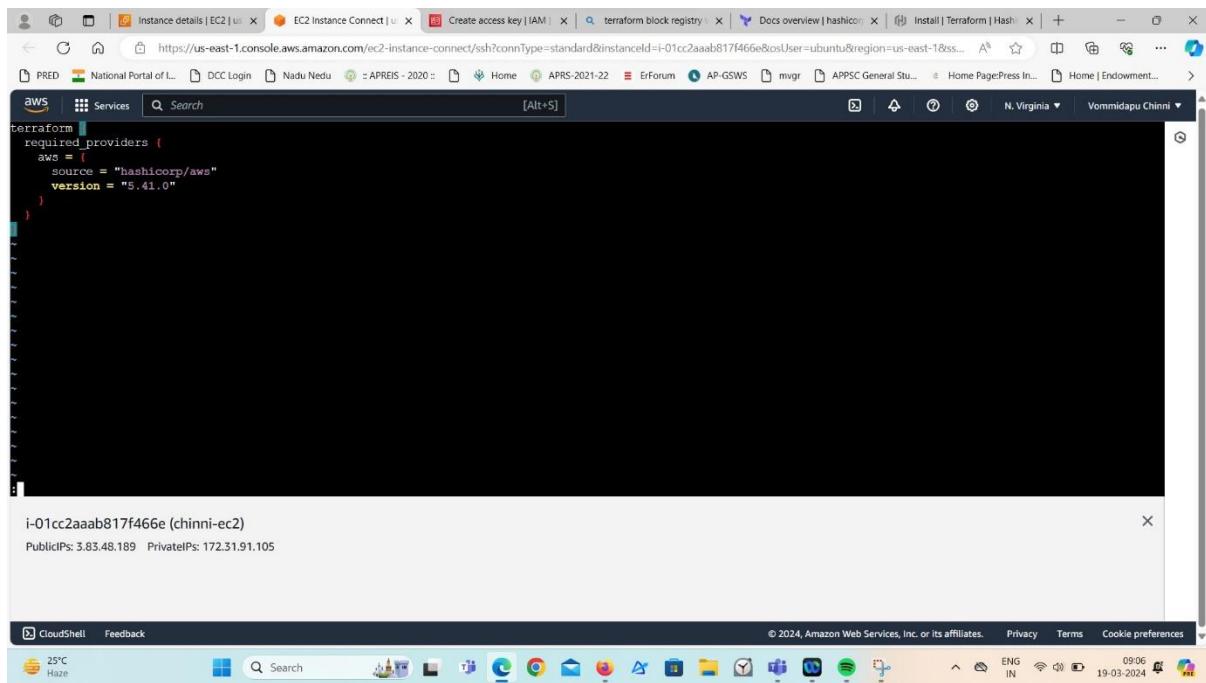
➤ Create terraform block using vi terraformblock.tf



```
aws Services Search [Alt+S]
root@ip-172-31-91-105:~# mkdir terraform
root@ip-172-31-91-105:~# cd terraform
root@ip-172-31-91-105:~/terraform# vi terraformblock.tf
root@ip-172-31-91-105:~/terraform#
```

i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

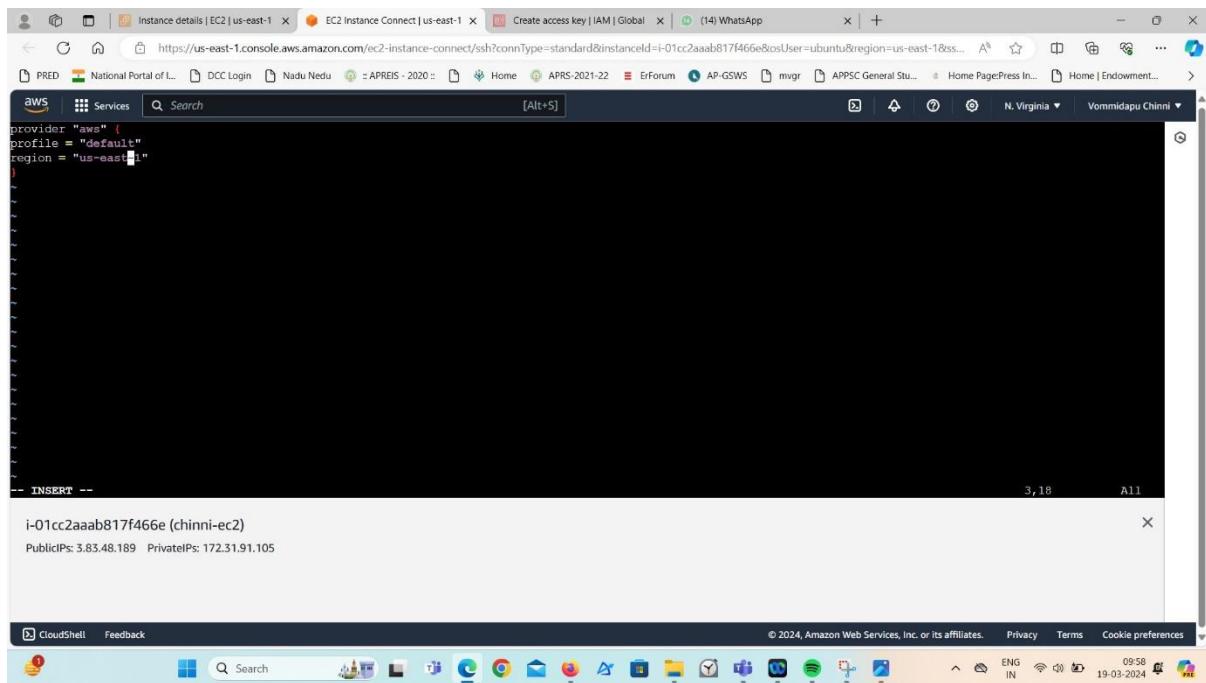
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
25°C 09:10 ENG IN 19-03-2024



```
aws
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.41.0"
    }
  }
```

i-01cc2aab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

➤ Create provider block using vi provider.tf



```
provider "aws" {
  profile = "default"
  region = "us-east-1"
}
```

-- INSERT --

i-01cc2aab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

➤ Create resource block using vi resource.tf

- For create an S3 bucket we take snippet from terraform registry.
- Creating a bucket using a name chinni46s3bucket.

```
aws
Services Search [Alt+S]
resource "aws_s3_bucket" "bucket" {
  bucket = "chinni1443bucket"
  tags = [
    { Name = "chinni-12345678" }
  ]
}

"resource.tf" 7L, 114B
i-01cc2aaab817f466e (chinni-ec2)
Public IPs: 3.83.48.189 Private IPs: 172.31.91.105

CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
25° ENG IN 09:33 19-03-2024
```

- Up to now we created three blocks, by using ls we can list the files in terraform directory.

```
aws
Services Search [Alt+S]
root@ip-172-31-91-105:~/terraform# vi provider.tf
root@ip-172-31-91-105:~/terraform# vi resource.tf
root@ip-172-31-91-105:~/terraform# ls
provider.tf resource.tf terraformblock.tf
root@ip-172-31-91-105:~/terraform#
```

i-01cc2aaab817f466e (chinni-ec2)
Public IPs: 3.83.48.189 Private IPs: 172.31.91.105

CloudShell Feedback
© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
25° ENG IN 09:55 19-03-2024

➤ **terraform init** (Preparing our working directory for other commands).

A screenshot of a terminal window titled "aws Services". The terminal shows the following command sequence:

```
root@ip-172-31-91-105:~/terraform# vi provider.tf
root@ip-172-31-91-105:~/terraform# vi resource.tf
root@ip-172-31-91-105:~/terraform# ls
provider.tf resource.tf terraformblock.tf
root@ip-172-31-91-105:~/terraform# terraform init

Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.41.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
root@ip-172-31-91-105:~/terraform#
```

The terminal window is part of a larger AWS CloudShell interface, which includes a status bar at the bottom showing "CloudShell Feedback" and a toolbar with various icons.

➤ **terraform validate** (Check whether the configuration is valid or not).

A screenshot of a terminal window titled "aws Services". The terminal shows the following command sequence:

```
root@ip-172-31-91-105:~/terraform# vi provider.tf
root@ip-172-31-91-105:~/terraform# vi resource.tf
root@ip-172-31-91-105:~/terraform# ls
provider.tf resource.tf terraformblock.tf
root@ip-172-31-91-105:~/terraform# terraform init

Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.41.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

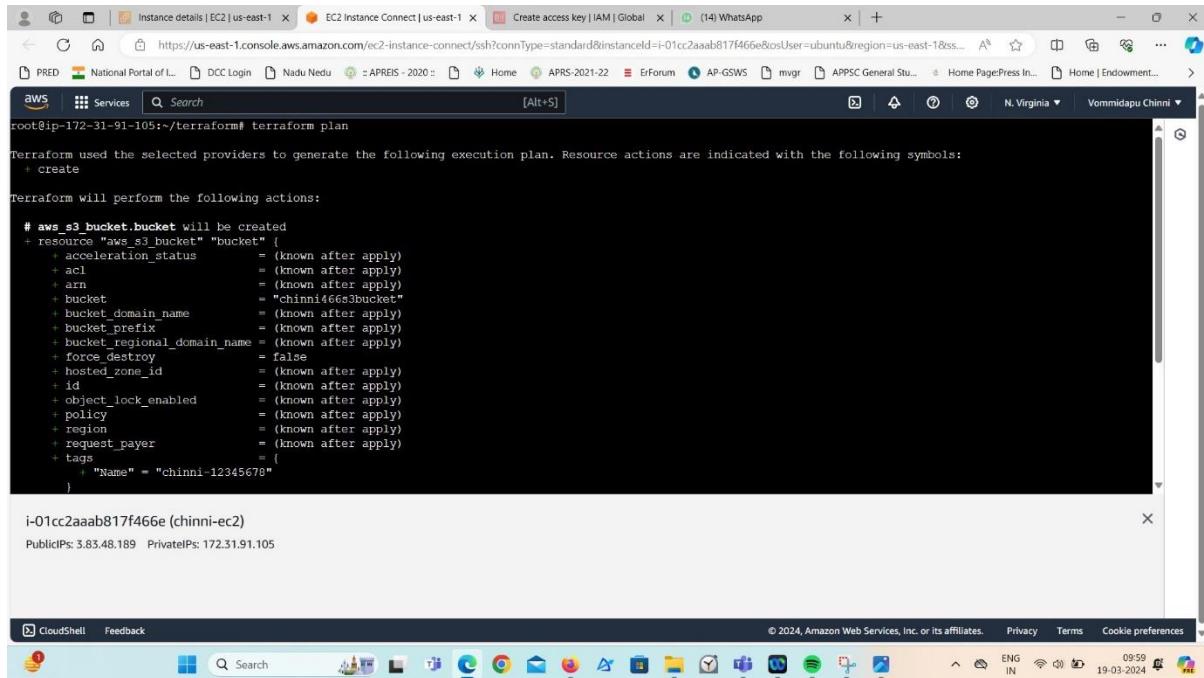
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
root@ip-172-31-91-105:~/terraform# terraform validate

Success! The configuration is valid.

root@ip-172-31-91-105:~/terraform#
```

The terminal window is part of a larger AWS CloudShell interface, which includes a status bar at the bottom showing "CloudShell Feedback" and a toolbar with various icons.

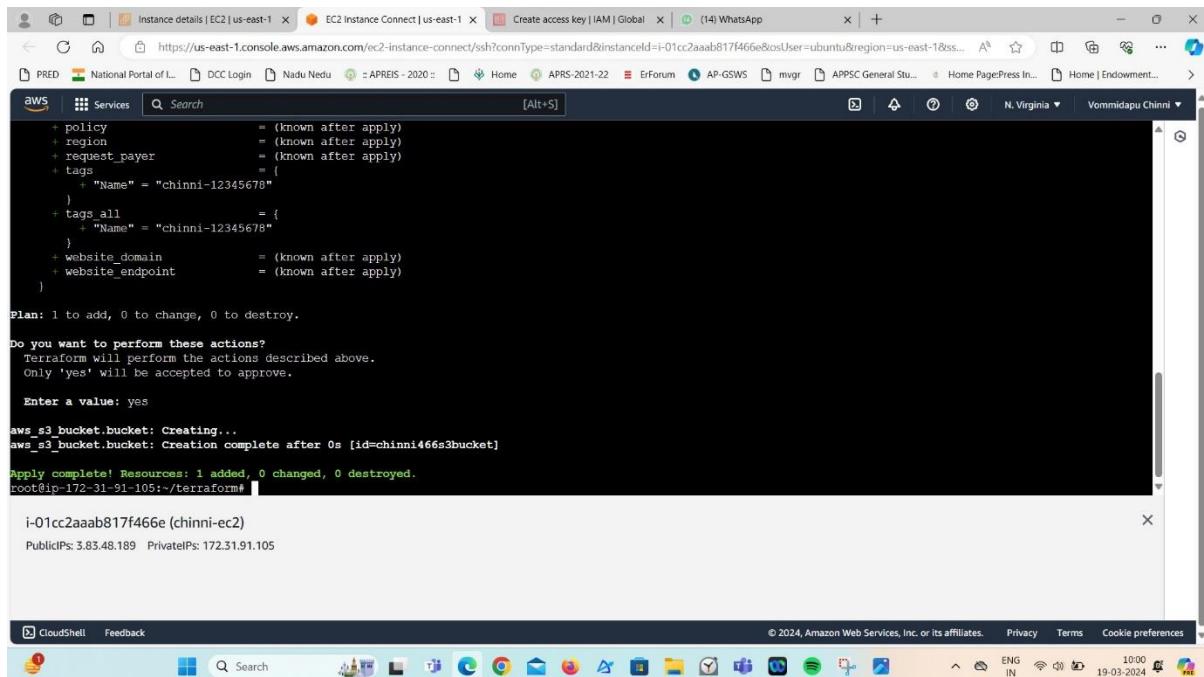
➤ terraform plan (show changes required by the current configuration).



```
Instance details | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | Create access key | IAM | Global | (14) WhatsApp | +  
[Alt+S]  
root@ip-172-31-91-105:~/terraform# terraform plan  
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:  
+ create  
Terraform will perform the following actions:  
  
# aws_s3_bucket.bucket will be created  
+ resource "aws_s3_bucket" "bucket"  
  + acceleration_status      = (known after apply)  
  + acl                      = (known after apply)  
  + arn                      = (known after apply)  
  + bucket                   = "chinni466s3bucket"  
  + bucket_domain_name       = (known after apply)  
  + bucket_prefix             = (known after apply)  
  + bucketRegionalDomainName = (known after apply)  
  + force_destroy            = false  
  + hostedZoneId             = (known after apply)  
  + id                       = (known after apply)  
  + objectLockEnabled        = (known after apply)  
  + policy                   = (known after apply)  
  + region                   = (known after apply)  
  + requestPayer              = (known after apply)  
  + tags                     = {  
    + "Name" = "chinni-12345678"  
  }  
  
i-01cc2aaab817f466e (chinni-ec2)  
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 09:59 19-03-2024

➤ terraform apply (create the infrastructure).



```
Instance details | EC2 | us-east-1 | EC2 Instance Connect | us-east-1 | Create access key | IAM | Global | (14) WhatsApp | +  
[Alt+S]  
root@ip-172-31-91-105:~/terraform# terraform apply  
+ policy          = (known after apply)  
+ region          = (known after apply)  
+ request_payer   = (known after apply)  
+ tags            = {  
  + "Name" = "chinni-12345678"  
}  
+ tags_all        = {  
  + "Name" = "chinni-12345678"  
}  
+ website_domain  = (known after apply)  
+ website_endpoint = (known after apply)  
  
Plan: 1 to add, 0 to change, 0 to destroy.  
  
Do you want to perform these actions?  
Terraform will perform the actions described above.  
Only 'yes' will be accepted to approve.  
  
Enter a value: yes  
  
aws_s3_bucket.bucket: Creating...  
aws_s3_bucket.bucket: Creation complete after 0s [id=chinni466s3bucket]  
  
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.  
root@ip-172-31-91-105:~/terraform#  
  
i-01cc2aaab817f466e (chinni-ec2)  
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 10:00 19-03-2024

- Open S3 dashboard.
- Click on buckets.
- We can see that bucket is created with a name chinni466s3bucket.

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with options like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings, Storage Lens (with sub-options like Dashboards, Storage Lens groups, and AWS Organizations settings), and a Feature spotlight. The main area is titled 'Amazon S3' and shows an 'Account snapshot' with a link to 'View Storage Lens dashboard'. Below that, it says 'General purpose buckets (1) [Info](#)' and 'Buckets are containers for data stored in S3.' There's a search bar labeled 'Find buckets by name'. A table lists the bucket details:

Name	AWS Region	Access	Creation date
chinni466s3bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	March 19, 2024, 10:00:14 (UTC+05:30)

- Give command like ls, we can see four files, that terraform.tfstate file contains that what was created in it.

```

+ request_payer      = (known after apply)
+ tags               = [
  + "Name" = "chinni-12345678"
]
+ tags_all           = [
  + "Name" = "chinni-12345678"
]
+ website_domain     = (known after apply)
+ website_endpoint   = (known after apply)
)

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

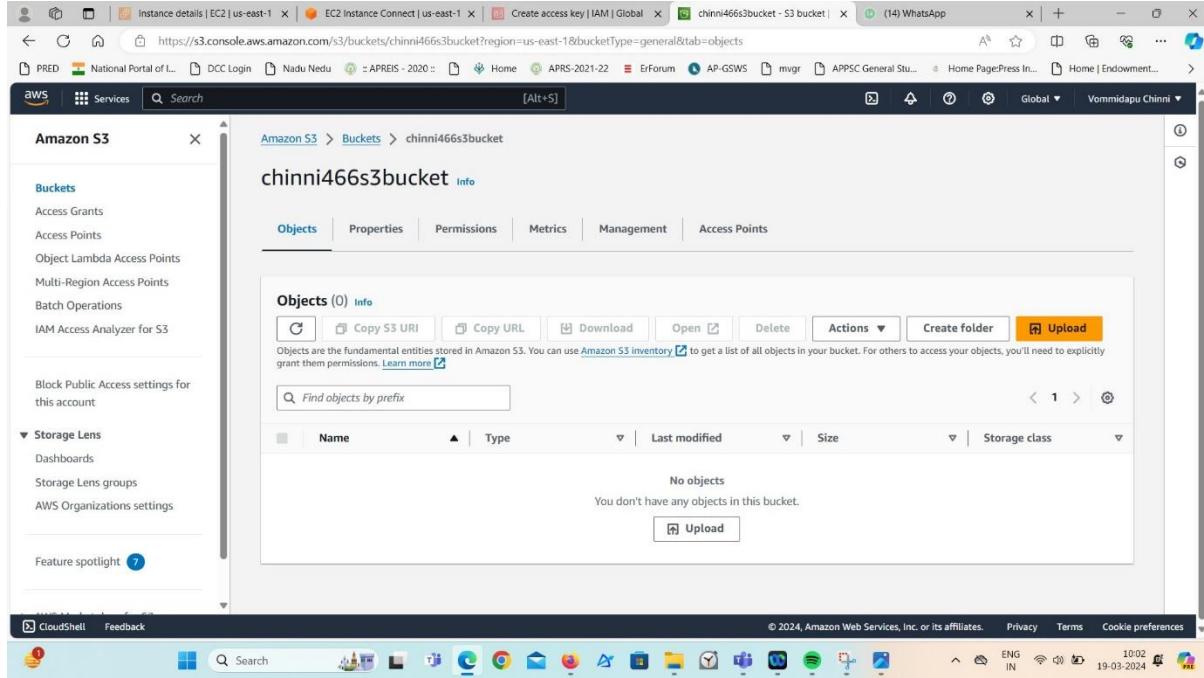
aws_s3_bucket.bucket: Creating...
aws_s3_bucket.bucket: Creation complete after 0s [id=chinni466s3bucket]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
root@ip-172-31-91-105:/terra# ls
provider.tf resource.tf terraform.tfstate terraformblock.tf
root@ip-172-31-91-105:/terra# i-01cc2aab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

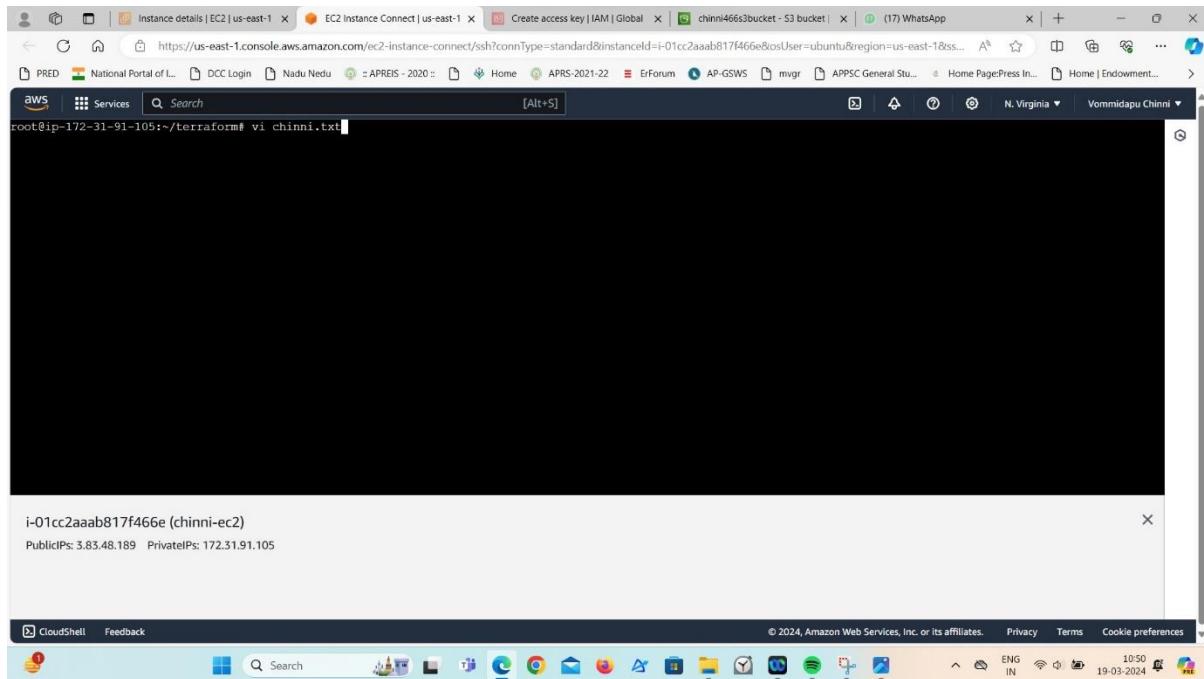
```

➤ Upload an object/file to the S3 bucket (chinni466s3bucket).

- Before that open chinni466s3bucket, there is nothing object/file in that bucket.



➤ Create a text file in that directory, by giving the name chinni.txt (vi chinni.txt).



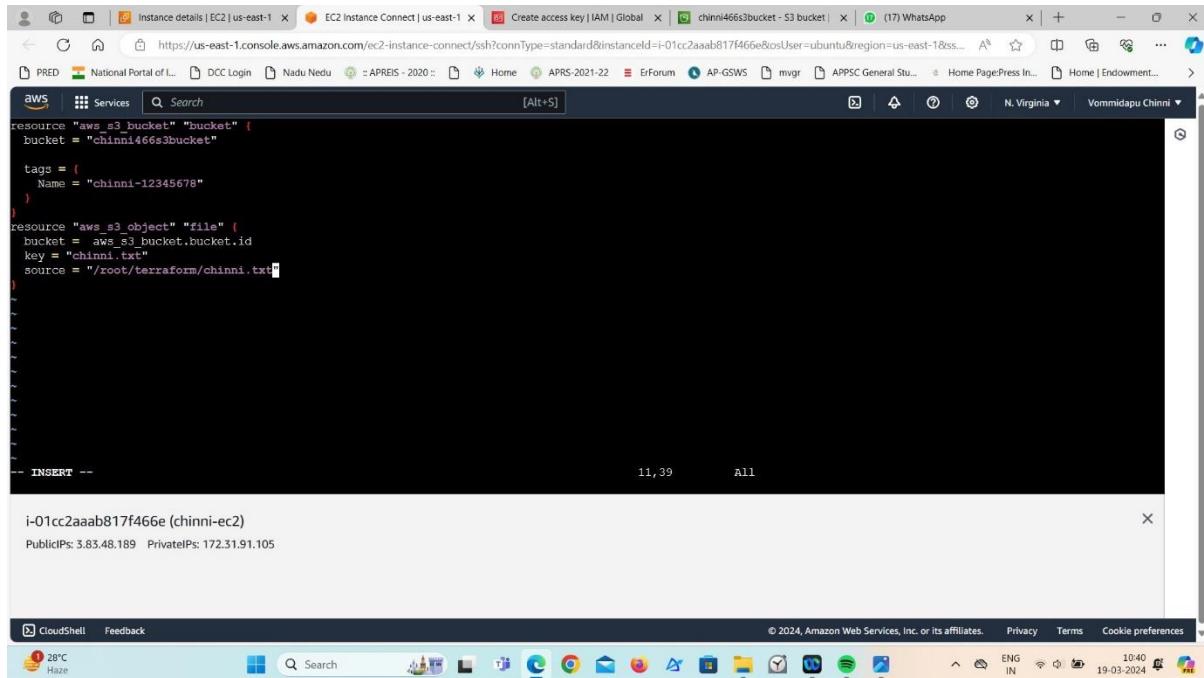
➤ In that enter the text (hello all).

```
hello all
```
-- INSERT --
i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

- Open vi resource.tf

```
root@ip-172-31-91-105:~/terraform# ls
provider.tf resource.tf terraform terraformblock.tf
root@ip-172-31-91-105:~/terraform# vi resource.tf
root@ip-172-31-91-105:~/terraform#
```

➤ Open terraform registry copy the snippet for S3 bucket object upload.



A screenshot of a web browser window. The address bar shows the URL: https://us-east-1.console.aws.amazon.com/ec2-instance-connect/shell?connType=standard&instanceId=i-01cc2aaab817f466e&osUser=ubuntu&region=us-east-1&ss... The browser title bar includes tabs for 'Instance details | EC2 | us-east-1', 'EC2 Instance Connect | us-east-1', 'Create access key | IAM | Global', 'chinni466s3bucket - S3 bucket', and '(17) WhatsApp'. Below the tabs, there's a navigation bar with links like 'PRED', 'National Portal of...', 'DCC Login', 'Nadu Nedu', 'APREIS - 2020', 'Home', 'APRS-2021-22', 'ErForum', 'AP-GSWS', 'mvgr', 'APPSC General Stu...', 'Home Page/Press In...', 'Home | Endowment...', 'N. Virginia', and 'Vommidapu Chinni'. The main content area displays a Terraform configuration file:

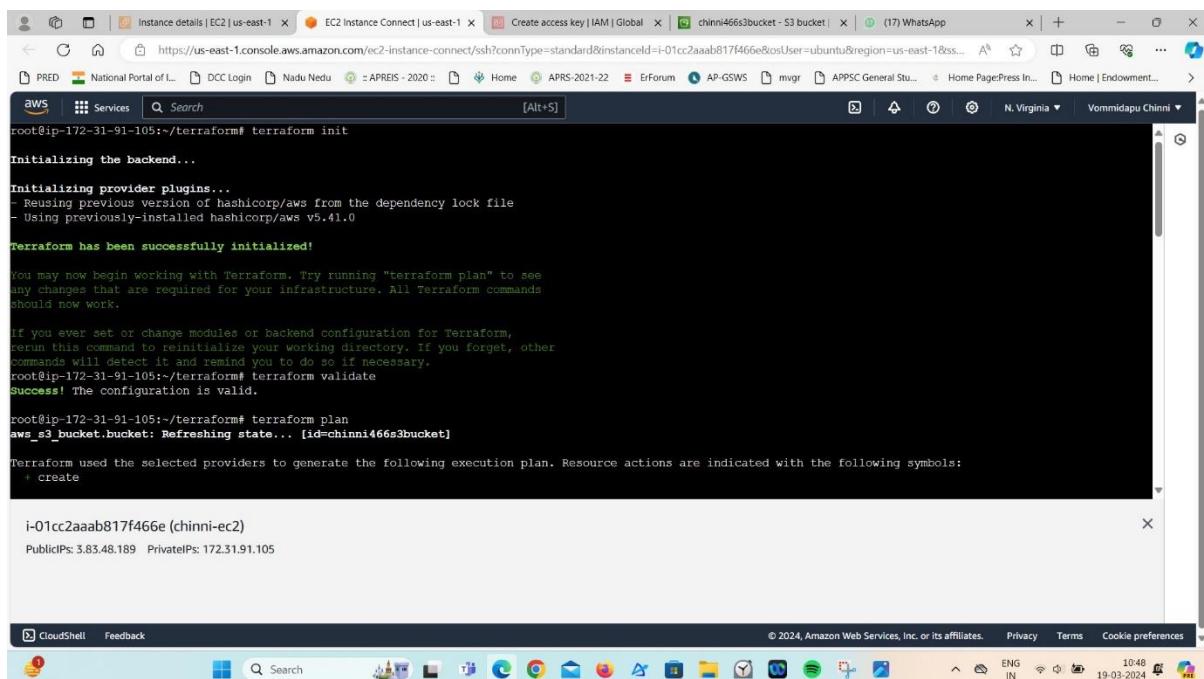
```
resource "aws_s3_bucket" "bucket" {
 bucket = "chinni466s3bucket"

 tags = {
 Name = "chinni-12345678"
 }
}

resource "aws_s3_object" "file" {
 bucket = aws_s3_bucket.bucket.id
 key = "chinni.txt"
 source = "/root/terraform/chinni.txt"
}
```

The status bar at the bottom indicates 'INSERT --' and shows the date and time as '11, 39' and 'All'. At the bottom of the browser window, there's a footer with 'CloudShell Feedback' and other system icons.

➤ terraform init and terraform validate.



A screenshot of a terminal window titled 'CloudShell' with the AWS logo. The command line shows the user running Terraform commands on an EC2 instance. The session starts with:

```
root@ip-172-31-91-105:~/terraform# terraform init
```

Output from 'terraform init':

```
Initializing the backend...
```

```
Initializing provider plugins...
Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.41.0
```

Output from 'Terraform has been successfully initialized!':

```
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
```

Instructions for reinitialization:

```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

```
root@ip-172-31-91-105:~/terraform# terraform validate
```

Output from 'Terraform validate':

```
Success! The configuration is valid.
```

```
root@ip-172-31-91-105:~/terraform# terraform plan
aws_s3_bucket.bucket: Refreshing state... [id=chinni466s3bucket]
```

Output from 'Terraform plan':

```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
```

The status bar at the bottom indicates 'INSERT --' and shows the date and time as '10:40' and '19-03-2024'. At the bottom of the terminal window, there's a footer with 'CloudShell Feedback' and other system icons.

## ➤ terraform plan

```
aws_s3_bucket.bucket: Refreshing state... [id=chinni466s3bucket]
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

aws_s3_object.file will be created
+ resource "aws_s3_object" "file" {
 acl = (known after apply)
 arn = (known after apply)
 bucket = "chinni466s3bucket"
 bucket_key_enabled = (known after apply)
 checksum_crc32 = (known after apply)
 checksum_crc32c = (known after apply)
 checksum_sh1 = (known after apply)
 checksum_sha256 = (known after apply)
 content_type = (known after apply)
 etag = (known after apply)
 force_destroy = false
 id = (known after apply)
 key = "chinni.txt"
 kms_key_id = (known after apply)
 server_side_encryption = (known after apply)
 source = "/root/terraform/chinni.txt"
}

i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```

## ➤ terraform apply --auto-approve (without asking yes it can approve it).

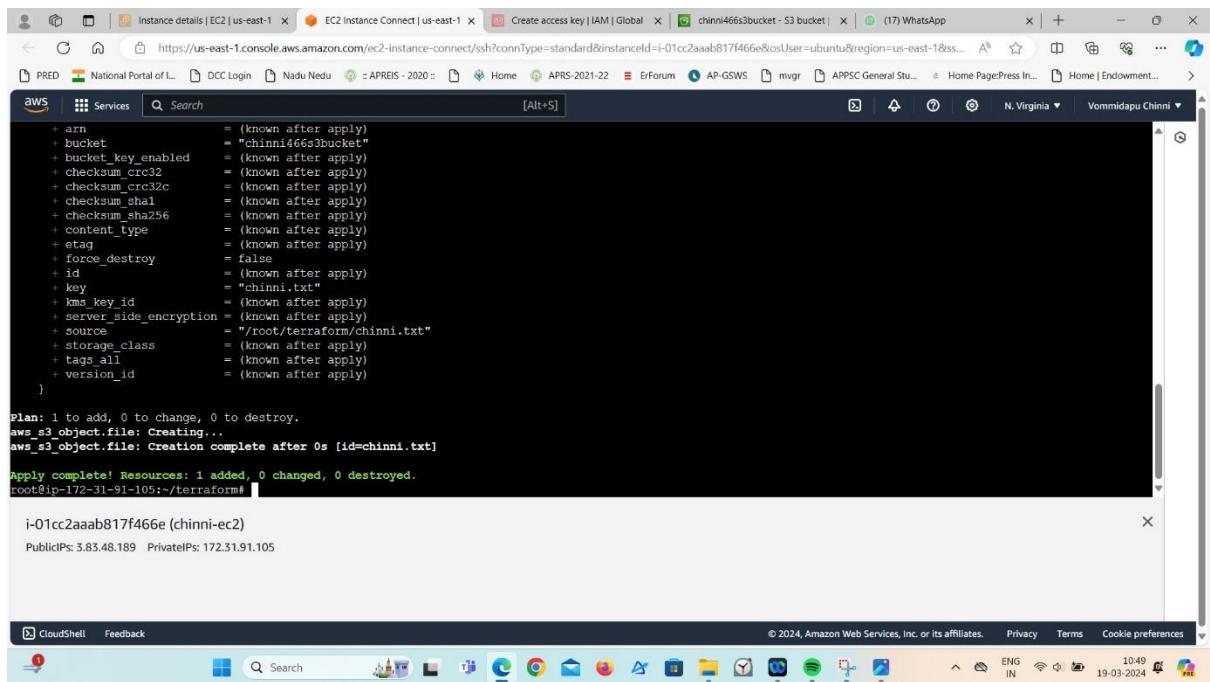
```
Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
root@ip-172-31-91-105:~/terraform# terraform apply --auto-approve
aws_s3_bucket.bucket: Refreshing state... [id=chinni466s3bucket]
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

aws_s3_object.file will be created
+ resource "aws_s3_object" "file" {
 acl = (known after apply)
 arn = (known after apply)
 bucket = "chinni466s3bucket"
 bucket_key_enabled = (known after apply)
 checksum_crc32 = (known after apply)
 checksum_crc32c = (known after apply)
 checksum_sh1 = (known after apply)
 checksum_sha256 = (known after apply)
 content_type = (known after apply)
 etag = (known after apply)
 force_destroy = false
 id = (known after apply)
 key = "chinni.txt"
 kms_key_id = (known after apply)
 server_side_encryption = (known after apply)
 source = "/root/terraform/chinni.txt"
}

i-01cc2aaab817f466e (chinni-ec2)
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105
```



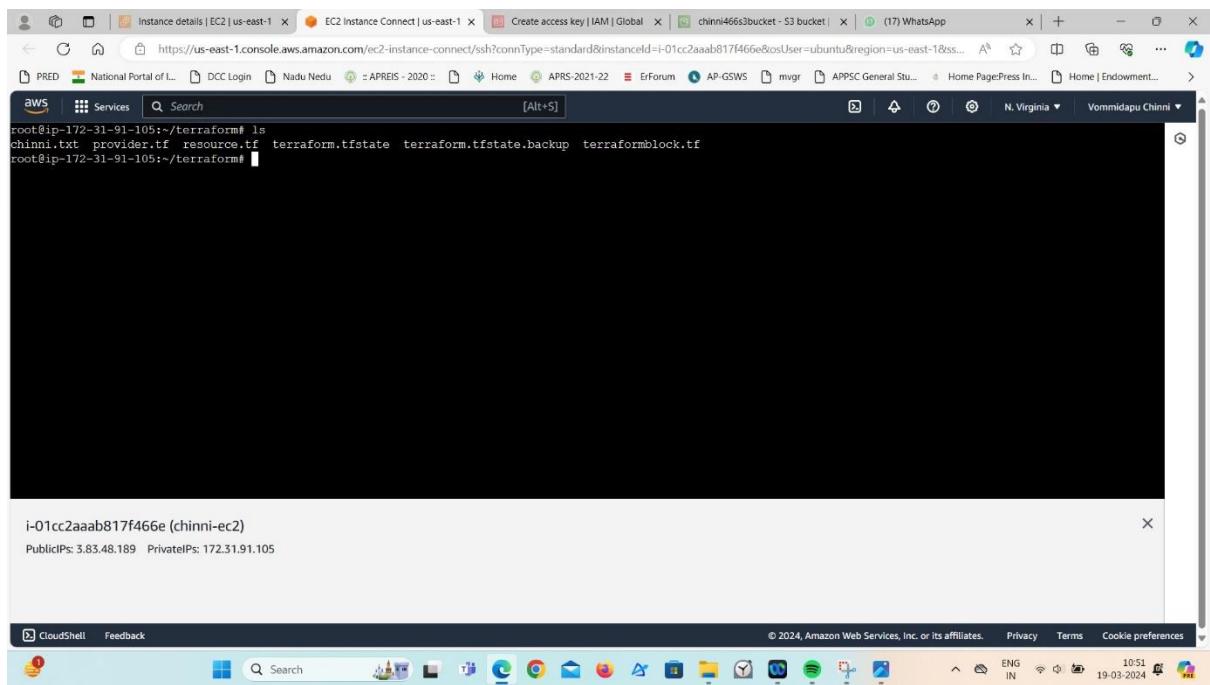
```
+ arn = (known after apply)
+ bucket = "chinni466s3bucket"
+ bucket_key_enabled = (known after apply)
+ checksum_crc32 = (known after apply)
+ checksum_crc32c = (known after apply)
+ checksum_shai = (known after apply)
+ checksum_sha256 = (known after apply)
+ content_type = (known after apply)
+ etag = (known after apply)
+ force_destroy = false
+ id = "chinni.txt"
+ kms_key_id = (known after apply)
+ server_side_encryption = (known after apply)
+ source = "/root/terraform/chinni.txt"
+ storage_class = (known after apply)
+ tags_all = (known after apply)
+ version_id = (known after apply)

Plan: 1 to add, 0 to change, 0 to destroy.
aws_s3_object.file: Creating...
aws_s3_object.file: Creation complete after 0s [id=chinni.txt]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
root@ip-172-31-91-105:~/terraform#
```

i-01cc2aaab817f466e (chinni-ec2)  
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

➤ Up to know what we created we can see using ls command



```
root@ip-172-31-91-105:~/terraform# ls
chinni.txt provider.tf resource.tf terraform.tfstate terraform.tfstate.backup terraformblock.tf
root@ip-172-31-91-105:~/terraform#
```

i-01cc2aaab817f466e (chinni-ec2)  
PublicIPs: 3.83.48.189 PrivateIPs: 172.31.91.105

- Open S3 dashboard, open chinni466s3bucket, we can see a object/file uploaded in it. What was created in the directory (chinni.txt).

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with various AWS services like EC2, IAM, and CloudWatch Metrics. The main area shows the 'Buckets' section with 'chinni466s3bucket' selected. Under the 'Objects' tab, a single file named 'chinni.txt' is listed. The file details are as follows:

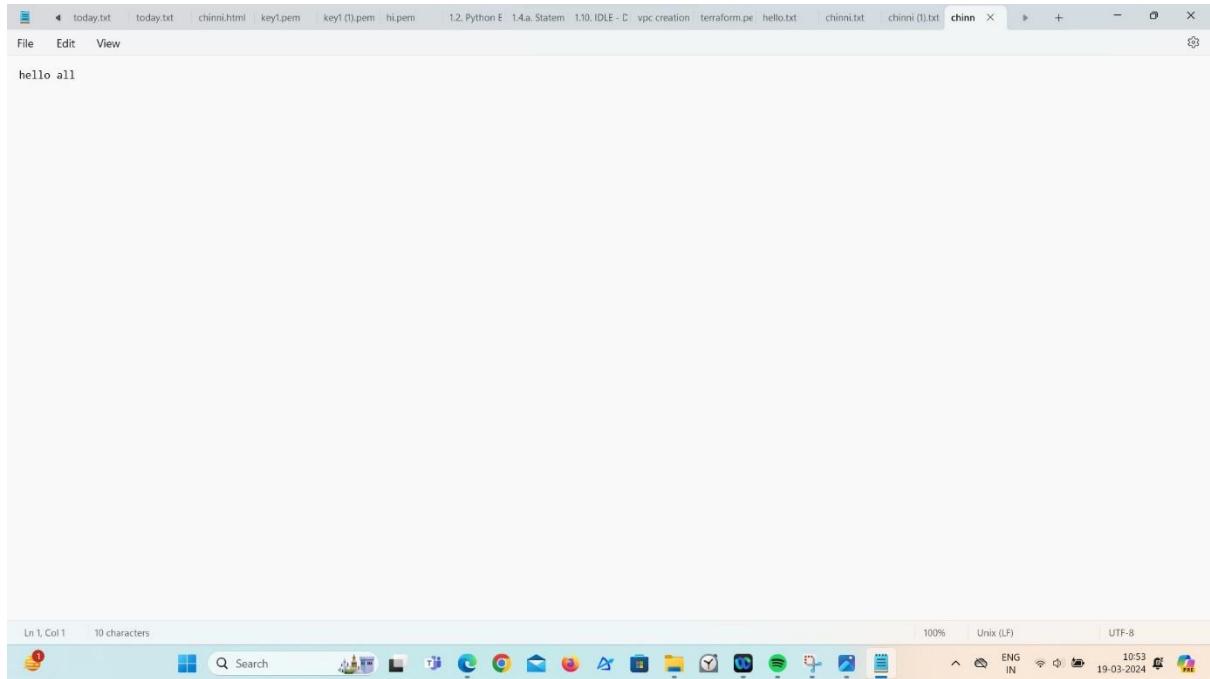
| Name       | Type | Last modified            | Size   | Storage class |
|------------|------|--------------------------|--------|---------------|
| chinni.txt | txt  | March 19, 2024, 10:47:07 | 10.0 B | Standard      |

- Open the file chinni.txt

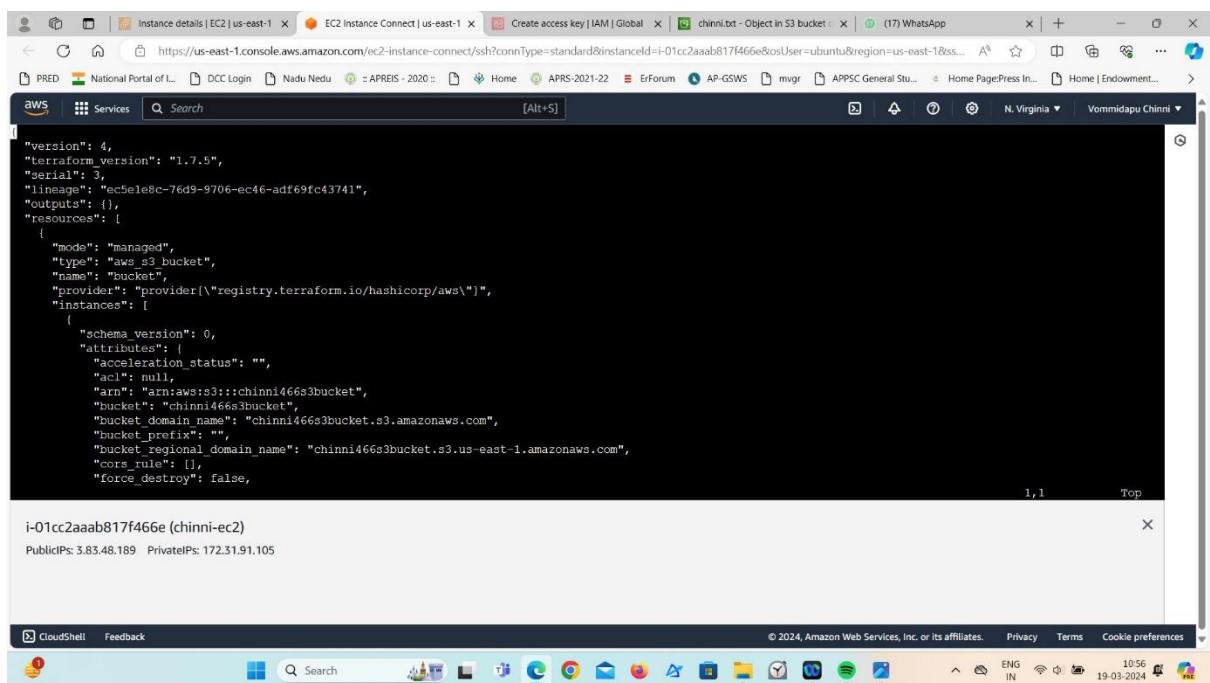
The screenshot shows the AWS S3 console interface, similar to the previous one but focusing on the file 'chinni.txt'. The 'Properties' tab is selected. The file details are as follows:

| Property      | Value                                |
|---------------|--------------------------------------|
| Owner         | vommidapuchinni                      |
| AWS Region    | US East (N. Virginia) us-east-1      |
| Last modified | March 19, 2024, 10:47:07 (UTC+05:30) |
| Size          | 10.0 B                               |
| Type          | txt                                  |
| Key           | chinni.txt                           |

- Click on open which is on the right side of the corner.
- We can see that chinni.txt is downloaded, open it, we can see that what is there in text file.



- We can open the terraform.tfstate file by using the command vi terraform.tfstate.



At last we have destroy by using the command `terraform destroy` (destroy previously created infrastructure).